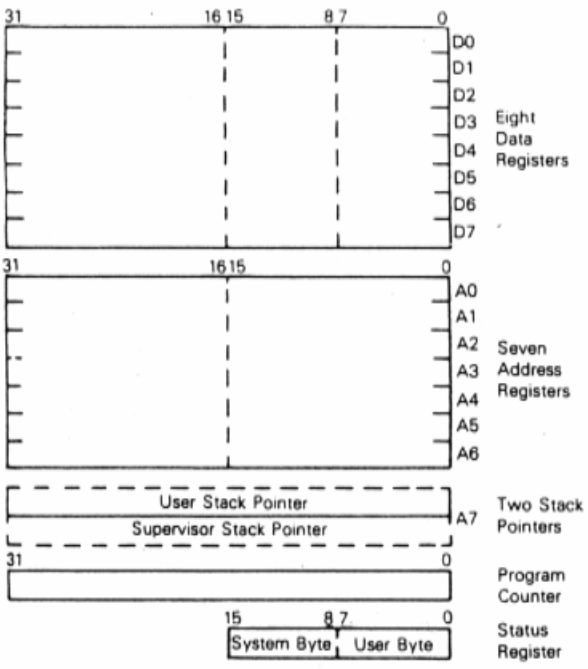
16-BIT MICROPROCESSING UNIT (MPU)

The R68000 microprocessor is designed for high performance where operational computation and versatility is required. The R68000 provides powerful mass-memory handling capability and architectural features designed to fit the broad range of 16-bit needs. The Rockwell family of 16-bit products also includes a wide range of peripherals that will allow complete system design and manufacture.

- 32-Bit Data and Address Registers
- 16 Megabyte Direct Addressing Range
- 56 Powerful Instruction Types
- Operations on Five Main Data Types
- Memory Mapped I/O
- 14 Addressing Modes

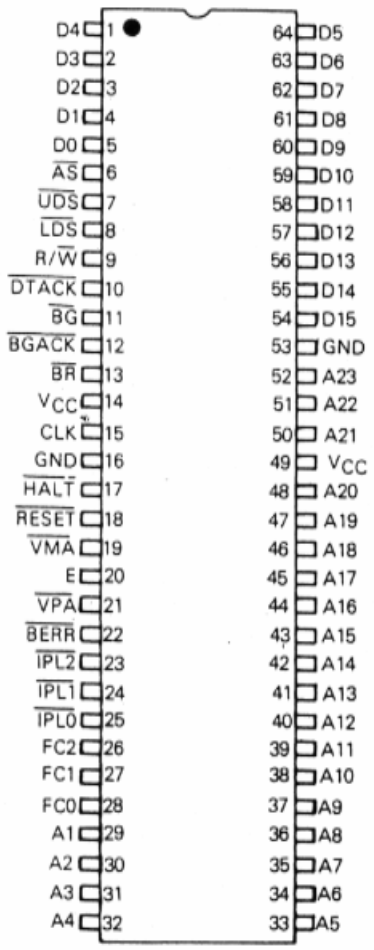
The R68000 offers seventeen 32-bit registers in addition to the 32-bit program counter and a 16-bit status register. The first eight registers (D0-D7) are used as data registers for byte (8-bit), word (16-bit), and long word (32-bit) data operations. The second set of seven registers (A0-A6) and the system stack pointer may be used as software stack pointers and base address registers. In addition, these registers may be used for word and long word address operations. All 17 registers may be used as index registers.



R68000 Registers

ORDERING INFORMATION

Order Number	Package Type	Frequency	Temperature Range
R68000C4	Ceramic	4 MHz	0°C to +70°C
R68000C6	Ceramic	6 MHz	0°C to +70°C
R68000C8	Ceramic	8 MHz	0°C to +70°C
R68000C10	Ceramic	10 MHz	0°C to +70°C



R68000 Pin Configuration

©Rockwell International Corporation 1982
All Rights Reserved
Printed in U.S.A.
Specifications subject to change without notice

Document No. 68650N01
Order No. 700
Rev. 3, November 1982

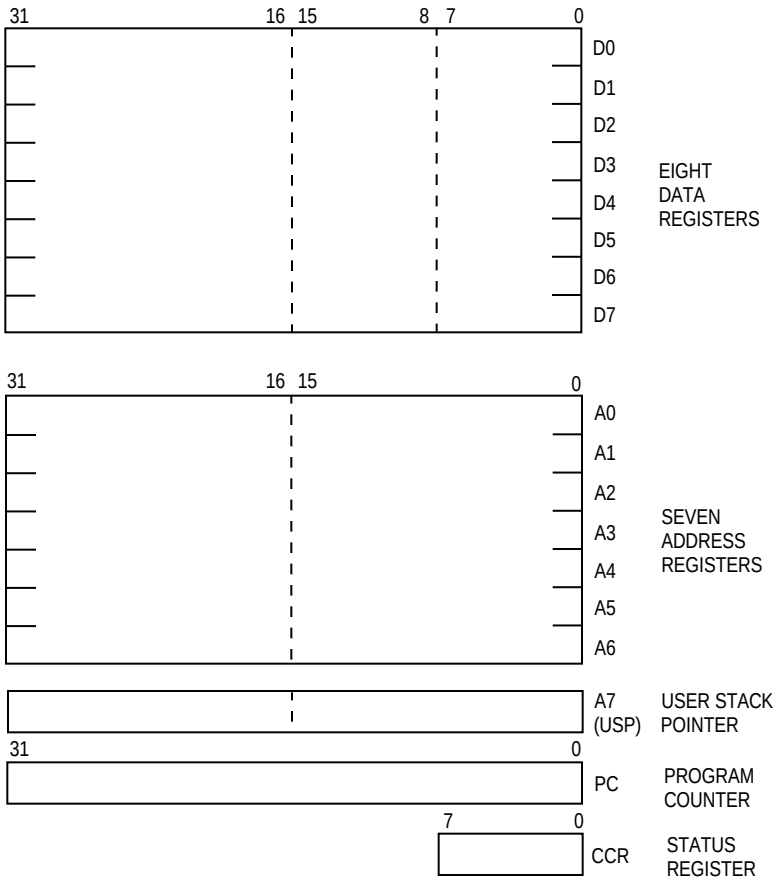


Figure 2-1. User Programmer's Model (MC68000/MC68HC000/MC68008/MC68010)

2.1.2 Supervisor Programmer's Model

The supervisor programmer's model consists of supplementary registers used in the supervisor mode. The M68000 MPUs contain identical supervisor mode register resources, which are shown in Figure 2-2, including the status register (high-order byte) and the supervisor stack pointer (SSP/A7').

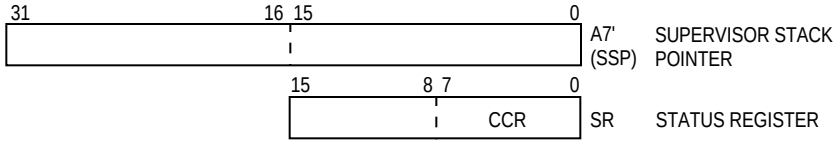


Figure 2-2. Supervisor Programmer's Model Supplement

The supervisor programmer's model supplement of the MC68010 is shown in Figure 2-3. In addition to the supervisor stack pointer and status register, it includes the vector base register (VRB) and the alternate function code registers (AFC).The VBR is used to determine the location of the exception vector table in memory to support multiple vector

tables. The SFC and DFC registers allow the supervisor to access user data space or emulate CPU space cycles.

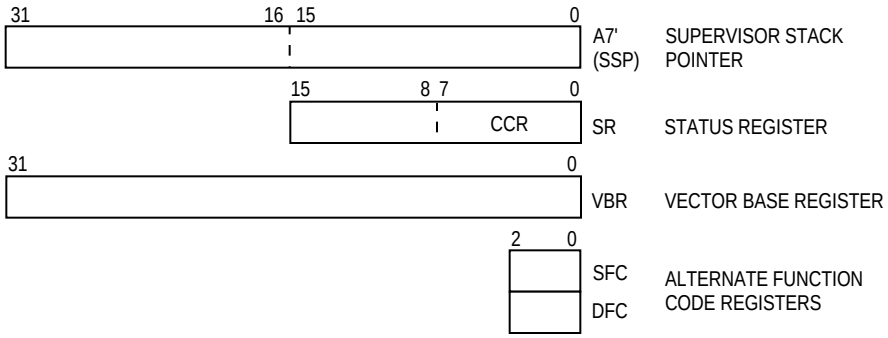


Figure 2-3. Supervisor Programmer's Model Supplement (MC68010)

2.1.3 Status Register

The status register (SR),contains the interrupt mask (eight levels available) and the following condition codes: overflow (V), zero (Z), negative (N), carry (C), and extend (X). Additional status bits indicate that the processor is in the trace (T) mode and/or in the supervisor (S) state (see Figure 2-4). Bits 5, 6, 7, 11, 12, and 14 are undefined and reserved for future expansion

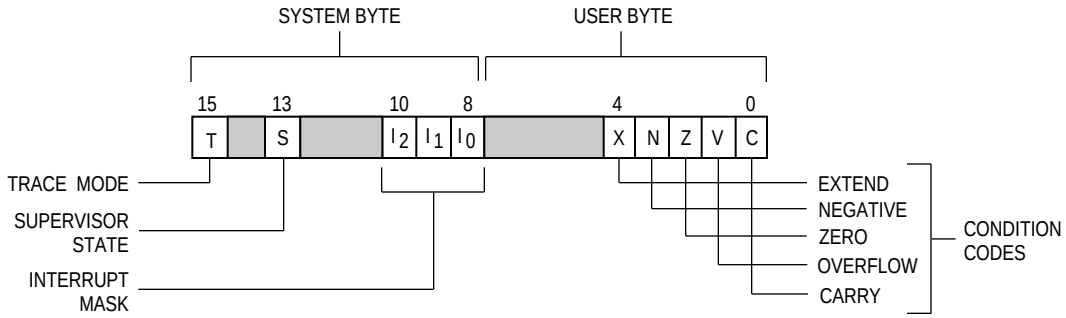


Figure 2-4. Status Register

2.2 DATA TYPES AND ADDRESSING MODES

The five basic data types supported are as follows:

1. Bits
2. Binary-Coded-Decimal (BCD) Digits (4 Bits)
3. Bytes (8 Bits)
4. Words (16 Bits)
5. Long Words (32 Bits)

SECTION 3
SIGNAL DESCRIPTION

This section contains descriptions of the input and output signals. The input and output signals can be functionally organized into the groups shown in Figure 3-1 (for the MC68000, the MC68HC000 and the MC68010), Figure 3-2 (for the MC68HC001), Figure 3-3 (for the MC68EC000), Figure 3-4 (for the MC68008, 48-pin version), and Figure 3-5 (for the MC68008, 52-pin version). The following paragraphs provide brief descriptions of the signals and references (where applicable) to other paragraphs that contain more information about the signals.

NOTE

The terms **assertion** and **negation** are used extensively in this manual to avoid confusion when describing a mixture of "active-low" and "active-high" signals. The term assert or assertion is used to indicate that a signal is active or true, independently of whether that level is represented by a high or low voltage. The term negate or negation is used to indicate that a signal is inactive or false.

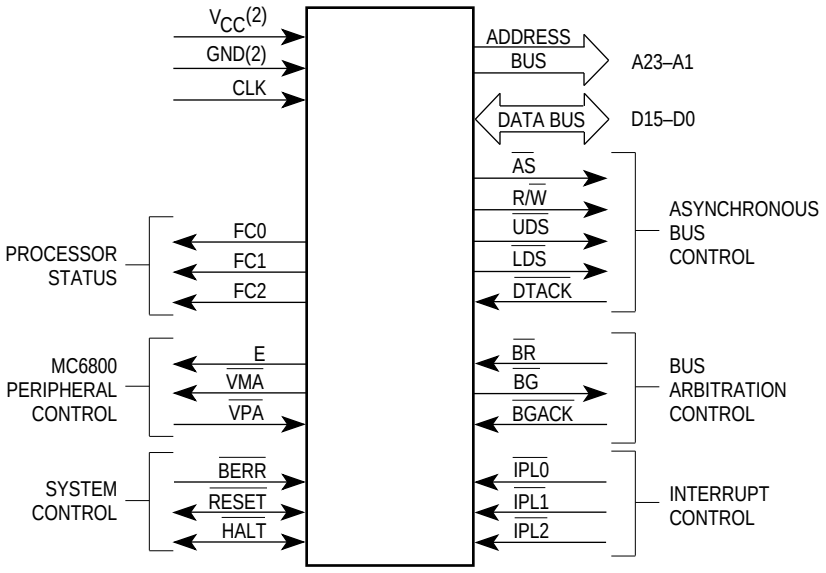


Figure 3-1. Input and Output Signals (MC68000, MC68HC000 and MC68010)

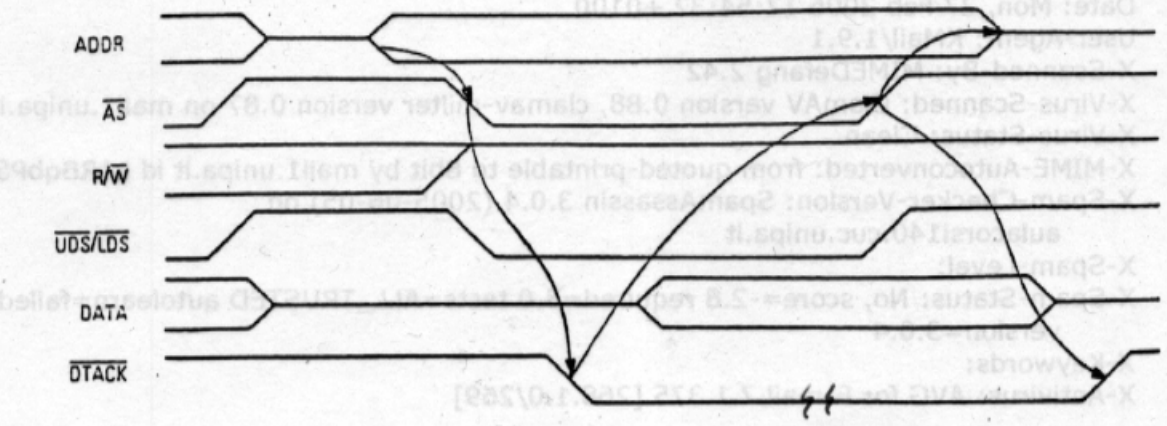


Fig 4.5. Ciclo di lettura relativo al microprocessore 68000. (dai data sheet della Motorola Inc.).

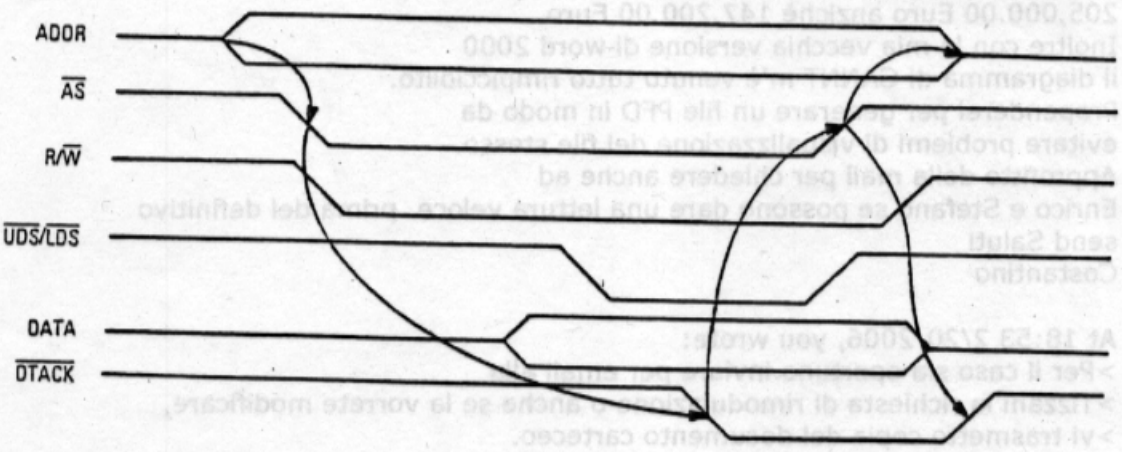


Fig. 4.6. Ciclo di scrittura relativo al microprocessore 68000. (dai data sheet della Motorola Inc.)

Table 3-3. Function Code Outputs

Function Code Output			
FC2	**FC1**	**FC0**	**Address Space Type**
Low	Low	Low	(Undefined, Reserved)
Low	Low	High	User Data
Low	High	Low	User Program
Low	High	High	(Undefined, Reserved)
High	Low	Low	(Undefined, Reserved)
High	Low	High	Supervisor Data
High	High	Low	Supervisor Program
High	High	High	CPU Space

3.9 CLOCK (CLK)

The clock input is a TTL-compatible signal that is internally buffered for development of the internal clocks needed by the processor. This clock signal is a constant frequency square wave that requires no stretching or shaping. The clock input should not be gated off at any time, and the clock signal must conform to minimum and maximum pulse-width times listed in **Section 10 Electrical Characteristics**.

3.10 POWER SUPPLY (V_{CC} and GND)

Power is supplied to the processor using these connections. The positive output of the power supply is connected to the V_{CC} pins and ground is connected to the GND pins.

3.11 SIGNAL SUMMARY

Table 3-4 summarizes the signals discussed in the preceding paragraphs.

Table 3-4. Signal Summary

Signal Name	Mnemonic	Input/Output	Active State	Hi-Z	
				On $\overline{HALT}$	On Bus Relinquish
Address Bus	A0–A23	Output	High	Yes	Yes
Data Bus	D0–D15	Input/Output	High	Yes	Yes
Address Strobe	$\overline{AS}$	Output	Low	No	Yes
Read/Write	R/$\overline{W}$	Output	Read-High Write-Low	No	Yes
Data Strobe	$\overline{DS}$	Output	Low	No	Yes
Upper and Lower Data Strobes	$\overline{UDS}$, $\overline{LDS}$	Output	Low	No	Yes
Data Transfer Acknowledge	$\overline{DTACK}$	Input	Low	No	No
Bus Request	$\overline{BR}$	Input	Low	No	No
Bus Grant	$\overline{BG}$	Output	Low	No	No
Bus Grant Acknowledge	$\overline{BGACK}$	Input	Low	No	No
Interrupt Priority Level	$\overline{IPL}$0, $\overline{IPL}$1, $\overline{IPL}$2	Input	Low	No	No
Bus Error	$\overline{BERR}$	Input	Low	No	No
Mode	MODE	Input	High	—	—
Reset	$\overline{RESET}$	Input/Output	Low	No*	No*
Halt	$\overline{HALT}$	Input/Output	Low	No*	No*
Enable	E	Output	High	No	No
Valid Memory Address	$\overline{VMA}$	Output	Low	No	Yes
Valid Peripheral Address	$\overline{VPA}$	Input	Low	No	No
Function Code Output	FC0, FC1, FC2	Output	High	No	Yes
Clock	CLK	Input	High	No	No
Power Input	V_{CC}	Input	—	—	—
Ground	GND	Input	—	—	—

*Open drain.

4. For an MC68010, return DTACK before data verification. If data is invalid, assert BERR on the next clock cycle (case 4).

Table 5-6. BERR and HALT Negation Results

Conditions of Termination in Table 4-4	Control Signal	Negated on Rising Edge of State			Results—Next Cycle
		N		N+2	
Bus Error	BERR HALT	• •	or or	• •	Takes bus error trap.
Rerun	BERR HALT	• •	or	•	Illegal sequence; usually traps to vector number 0.
Rerun	BERR HALT	•		•	Reruns the bus cycle.
Normal	BERR HALT	• •	or	•	May lengthen next cycle.
Normal	BERR HALT	•	or	• none	If next cycle is started, it will be terminated as a bus error.

• = Signal is negated in this bus state.

5.7 ASYNCHRONOUS OPERATION

To achieve clock frequency independence at a system level, the bus can be operated in an asynchronous manner. Asynchronous bus operation uses the bus handshake signals to control the transfer of data. The handshake signals are AS, UDS, LDS, DS (MC68008 only), DTACK, BERR, HALT, AVEC (MC68EC000 only), and VPA (only for M6800 peripheral cycles). AS indicates the start of the bus cycle, and UDS, LDS, and DS signal valid data for a write cycle. After placing the requested data on the data bus (read cycle) or latching the data (write cycle), the slave device (memory or peripheral) asserts DTACK to terminate the bus cycle. If no device responds or if the access is invalid, external control logic asserts BERR, or BERR and HALT, to abort or retry the cycle. Figure 5-31 shows the use of the bus handshake signals in a fully asynchronous read cycle. Figure 5-32 shows a fully asynchronous write cycle.

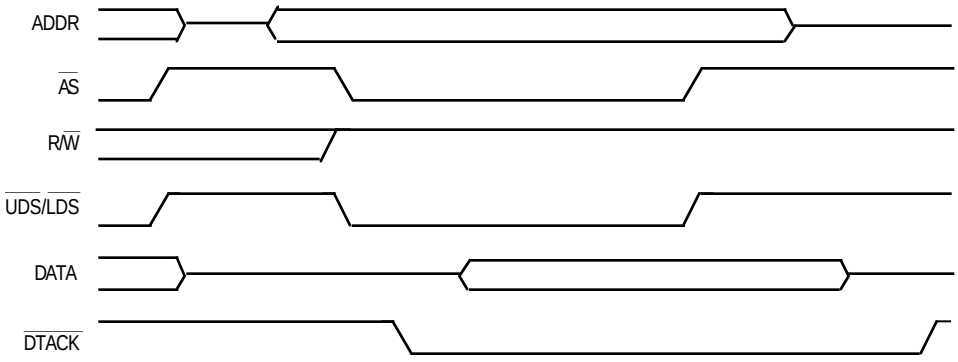


Figure 5-31. Fully Asynchronous Read Cycle

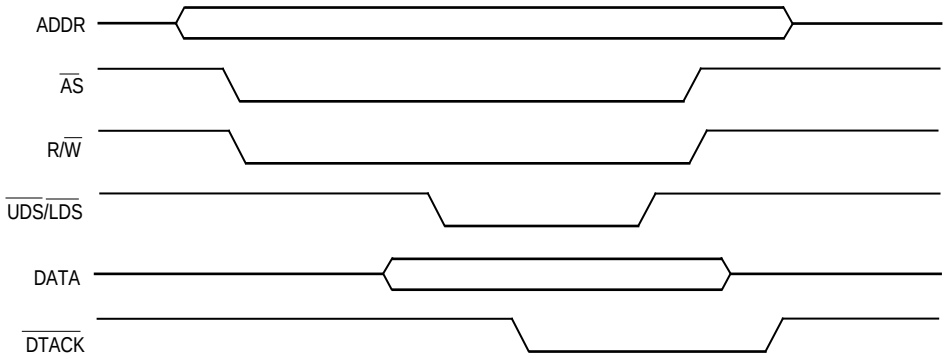


Figure 5-32. Fully Asynchronous Write Cycle

In the asynchronous mode, the accessed device operates independently of the frequency and phase of the system clock. For example, the MC68681 dual universal asynchronous receiver/transmitter (DUART) does not require any clock-related information from the bus master during a bus transfer. Asynchronous devices are designed to operate correctly with processors at any clock frequency when relevant timing requirements are observed.

A device can use a clock at the same frequency as the system clock (e.g., 8, 10, or 12.5, 16, and 20MHz), but without a defined phase relationship to the system clock. This mode of operation is pseudo-asynchronous; it increases performance by observing timing parameters related to the system clock frequency without being completely synchronous with that clock. A memory array designed to operate with a particular frequency processor but not driven by the processor clock is a common example of a pseudo-asynchronous device.

The designer of a fully asynchronous system can make no assumptions about address setup time, which could be used to improve performance. With the system clock frequency known, the slave device can be designed to decode the address bus before recognizing an address strobe. Parameter #11 (refer to **Section 10 Electrical Characteristics**) specifies the minimum time before address strobe during which the address is valid.

In a pseudo-asynchronous system, timing specifications allow $\overline{\text{DTACK}}$ to be asserted for a read cycle before the data from a slave device is valid. The length of time that $\overline{\text{DTACK}}$ may precede data is specified as parameter #31. This parameter must be met to ensure the validity of the data latched into the processor. No maximum time is specified from the assertion of $\overline{\text{AS}}$ to the assertion of $\overline{\text{DTACK}}$. During this unlimited time, the processor inserts wait cycles in one-clock-period increments until $\overline{\text{DTACK}}$ is recognized. Figure 5-33 shows the important timing parameters for a pseudo-asynchronous read cycle.

Table 6-2. Exception Vector Assignment

Vectors Numbers		Address		Space[6]	Assignment
Hex	Decimal	Dec	Hex		
0	0	0	000	SP	Reset: Initial SSP[2]
1	1	4	004	SP	Reset: Initial PC[2]
2	2	8	008	SD	Bus Error
3	3	12	00C	SD	Address Error
4	4	16	010	SD	Illegal Instruction
5	5	20	014	SD	Zero Divide
6	6	24	018	SD	CHK Instruction
7	7	28	01C	SD	TRAPV Instruction
8	8	32	020	SD	Privilege Violation
9	9	36	024	SD	Trace
A	10	40	028	SD	Line 1010 Emulator
B	11	44	02C	SD	Line 1111 Emulator
C	12[1]	48	030	SD	(Unassigned, Reserved)
D	13[1]	52	034	SD	(Unassigned, Reserved)
E	14	56	038	SD	Format Error[5]
F	15	60	03C	SD	Uninitialized Interrupt Vector
10–17	16–23[1]	64	040	SD	(Unassigned, Reserved)
		92	05C		—
18	24	96	060	SD	Spurious Interrupt[3]
19	25	100	064	SD	Level 1 Interrupt Autovector
1A	26	104	068	SD	Level 2 Interrupt Autovector
1B	27	108	06C	SD	Level 3 Interrupt Autovector
1C	28	112	070	SD	Level 4 Interrupt Autovector
1D	29	116	074	SD	Level 5 Interrupt Autovector
1E	30	120	078	SD	Level 6 Interrupt Autovector
1F	31	124	07C	SD	Level 7 Interrupt Autovector
20–2F	32–47	128	080	SD	TRAP Instruction Vectors[4]
		188	0BC		—
30–3F	48–63[1]	192	0C0	SD	(Unassigned, Reserved)
		255	0FF		—
40–FF	64–255	256	100	SD	User Interrupt Vectors
		1020	3FC		—

NOTES:

1. Vector numbers 12, 13, 16–23, and 48–63 are reserved for future enhancements by Motorola. No user peripheral devices should be assigned these numbers.
2. Reset vector (0) requires four words, unlike the other vectors which only require two words, and is located in the supervisor program space.
3. The spurious interrupt vector is taken when there is a bus error indication during interrupt processing.
4. TRAP #n uses vector number 32+ n.
5. MC68010 only. This vector is unassigned, reserved on the MC68000 and MC68008.
6. SP denotes supervisor program space, and SD denotes supervisor data space.

MC68000 instructions

- **Instruction format is**

```
<label> opcode<.field> <operands> <;comments>
```

  - `<label>` pointer to the instruction's memory location
  - `opcode` operation code (i.e., MOVE, ADD)
  - `<.field>` defines width of operands (B,W,L)
  - `<operands>` data used in the operation
  - `<;comments>` for program documentation

- **Examples**

	Instruction		RTL
	MOVE.W	#100,D0	[D0]←100
	MOVE.W	100,D0	[D0]←[M(100)]
	ADD.W	D0,D1	[D1]←[D1]+[D0]
	MOVE.W	D1,100	[M(100)]←[D1]
data	DC.B	20	[data] ←20
	BRA	label	[PC] ←label

Instruction operands

- **Operands can be**
  - registers
  - constants
  - memory addresses
- **Basic addressing modes (to be expanded)**
  - D_n: data register direct `MOVE.L D0, D1`
  - A_n: register indirect `MOVE.L (A0), D1`
  - #n: immediate `MOVE.L #10, D1`
  - n: absolute `MOVE.L $08FF00, D1`
- **Immediate operands can be specified in several formats**
  - Hexadecimal: prefixed by $
  - Octal: prefixed by @
  - Decimal: prefixed by & (or nothing)
  - Binary: prefixed by %
  - ASCII: within single quotes 'abc'

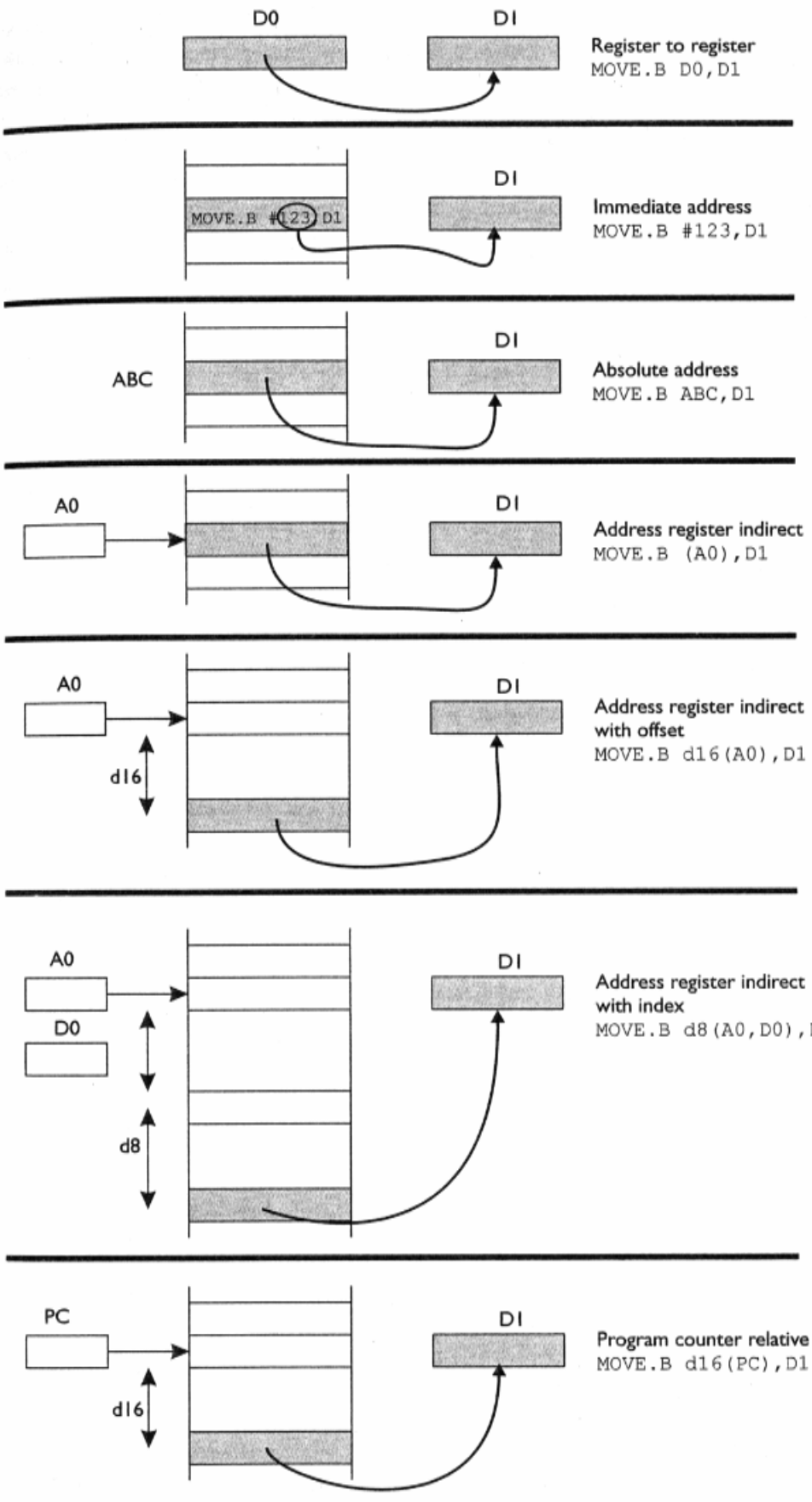


Fig. 6.25 Summary of the 68K's addressing modes.